

Non-Myopic Target Tracking Strategies for State-Dependent Noise

Zhongshun Zhang and Pratap Tokekar

Abstract—We study the problem of devising a closed-loop strategy to control the position of a robot that is tracking a possibly moving target. The robot is capable of obtaining noisy measurements of the target’s position. The key idea in active target tracking is to choose control laws that drive the robot to measurement locations that will reduce the uncertainty in the target’s position. The challenge is that measurement uncertainty often is a function of the (unknown) relative positions of the target and the robot. Consequently, a closed-loop control policy is desired which can map the current estimate of the target’s position to an optimal control law for the robot.

Our main contribution is to devise a closed-loop control policy for target tracking that plans for a sequence of control actions, instead of acting greedily. We consider scenarios where the noise in measurement is a function of the state of the target. We seek to minimize the maximum uncertainty (trace of the posterior covariance matrix) over all possible measurements. We exploit the structural properties of a Kalman Filter to build a policy tree that is orders of magnitude smaller than naive enumeration while still preserving optimality guarantees. We show how to obtain even more computational savings by relaxing the optimality guarantees. The resulting algorithms are evaluated through simulations.

I. INTRODUCTION

Tracking a moving, possibly adversarial target is a fundamental problem in robotics and has long been a subject of study [1]–[6]. Target tracking finds applications in many areas such as surveillance [7], telepresence [8], assisted living [9], and habitat monitoring [10], [11]. Target tracking refers to broadly two classes of problems: (i) estimating the position of the target using noisy sensor measurements; and (ii) actively controlling the sensor position to improve the performance of the estimator. The second problem is distinguished as *active* target tracking and is the subject of study of this paper.

The main challenge in active target tracking is that the value of future measurement locations can be a function of the unknown target state. Take as example, a simple instance of estimating the unknown position of a stationary target where the measurement noise is a function of the distance between the robot and the target. If the true location of the target were known, the robot would always choose a control sequence that drives it closer to the target. Since, in practice, the true target location is unknown, we cannot determine such a control sequence exactly. A possible strategy in this case would be to plan with respect to the probability distribution of the target. However, the probability distribution itself will evolve as a function of the actual measurement

values. This becomes even more challenging if the target is mobile. We use a Kalman Filter (KF) to estimate the state of a moving target with a possibly state-dependent measurement model where the measurement noise is a function of the distance between the robot and the target.

When planning non-myopically (*i.e.*, for multiple steps in the future), one can enumerate all possible future measurements in the form of a tree. In particular, a minimax tree can be used to find the optimal (in the *min-max* sense) control policy for actively tracking a target [12]. The size of the minmax tree grows exponentially with the time horizon. The tree can be pruned using $\alpha - \beta$ pruning [13]. Our main contribution is to show how the properties of a KF can be exploited to prune a larger number of nodes without losing optimality. In doing so, we extend the pruning techniques first proposed by Vitus et al. [14] for linear systems. Using a min max tree, we generalize these results to a state-dependent, time-variant dynamical systems. Our pruning techniques allow us to trade-off the size of the tree (equivalently, computation time) with the performance guarantees of the algorithm. We demonstrate this effect in simulations.

The rest of the paper is organized as follows. We start with the related work in Section II. The problem formulation is presented in Section III. Our main algorithm is presented in Section IV. We validate the algorithm through simulations described in Section VI. Finally, we conclude with a brief discussion of future work in Section VII.

II. RELATED WORK

The target tracking problem has been studied under various settings. Bar-Shalom et al. [1] present many of the commonly-used estimation techniques in target tracking. The five-part survey by Li and Jilkov [2]–[6] covers commonly-used control and maneuvering techniques for active target tracking. In the rest of the section, we discuss works most closely related to our formulation.

Vitus et al. [14] presented an algorithm that computes the optimal scheduling of measurements for a linear dynamical system. Their formulation does not directly model a target tracking problem. Instead, the goal is to track a linear dynamical system using a set of sensors such that one sensor can be activated at any time instance. The posterior covariance in estimating a linear system in a Kalman filter depends on the prior covariance and sensor variance but not on the actual measurement values (unlike the case in non-linear systems). Thus, one can build a search tree enumerating all possible sensor selections and choosing the one that minimizes the final covariance. The main contribution of Vitus et al. was

The authors are with the Department of Electrical & Computer Engineering, Virginia Tech, USA. {zszzhang, tokekar}@vt.edu.

This material is based upon work supported by the National Science Foundation under Grant #1566247.

to present a pruning technique to reduce the size of the tree while still preserving optimality.

Atanasov et al. [15] extended this result to active target tracking with a single robot. A major contribution was to show that robot trajectories that are nearby in space can be pruned away (under certain conditions), leading to further computational savings. This was based on a linear system assumption. In this paper, we build on these works and make progress towards generalizing the solution for state-dependent observation systems.

A major bottleneck for planning for state-dependent observation systems is that the future covariance is no longer independent of the actual measurement values, as was the case in linear state-independent systems. The covariance update equations use the linear models and as such will depend on the state estimate and the measurements. Thus, the search tree will have to include all possible measurement values and not just all possible measurement locations. Furthermore, finding an optimal path is no longer sufficient. Instead one must find an optimal policy that prescribes the optimal set of actions for all possible measurements. We show how to use a min max tree to find such an optimal policy while at the same time leveraging the computational savings that hold for the linear case.

III. PROBLEM FORMULATION

We assume that the position of the robot is known accurately using onboard sensors (*e.g.*, GPS, laser, IMU, cameras). The motion model of the robot is given by:

$$X_r(t+1) = f(X_r(t), u(t)) \quad (1)$$

where $X_r = [x_r(t), y_r(t)]^T \in \mathbb{R}^2$ is the position of robot and $u(t) \in \mathcal{U}$ is the control input at time t . \mathcal{U} is a finite space of control inputs. We assume there are n actions available as control inputs at any time:

$$\mathcal{U}(t) = \{u_1(t), u_2(t), \dots, u_n(t)\}.$$

The robot is mounted with a sensor that is capable of obtaining a measurement of the target. We assume that the target's motion model is given by:

$$X_o(t+1) = C_t X_o(t) + v(t) \quad (2)$$

where, $X_o(t) = [x_o(t), y_o(t)]^T$ is the 2-dimensional position of the target and $v(t)$ is Gaussian noise of known covariance.

The task of the robot is to track the target using its noisy measurements. The measurements, $Z(t)$, can be a nonlinear function of the states of the target and robot:

$$Z(t) = H(X_r(t))X_o(t) + \omega(X_r(t), X_o(t)) \quad (3)$$

The measurement noise, $\omega(t)$, is a Gaussian whose variance depends on the distance between the robot and the target:

$$\omega(X_r(t), X_o(t)) \sim N(0, \delta_1^2 + \delta_2^2 d(X_r(t), X_o(t))) \quad (4)$$

where,

$$d(X_r(t), X_o(t)) = \begin{cases} C, & \|X_r(t) - X_o(t)\|_2 > \mathcal{B} \\ \frac{C\|X_r(t) - X_o(t)\|_2}{\mathcal{B}}, & \|X_r(t) - X_o(t)\|_2 \leq \mathcal{B} \end{cases} \quad (5)$$

When the true distance between the robot and target is within \mathcal{B} , we assume that measurement noise is proportional to the true distance. When the distance is greater than \mathcal{B} , the variance is assumed to be a constant maximum value of C .

The estimated position and the covariance matrix of the target at time t , $\hat{X}_o(t)$ and $\hat{\Sigma}_o(t)$, is given by the Kalman Filter. The uncertainty in the estimate of the target's position is measured by the trace of the covariance matrix. The problem considered in this paper can be formally stated as follows.

Problem: Given an initial robot position $X_r(0)$ and an initial target estimate $[\hat{X}_o(0), \hat{\Sigma}_o(0)]$, find a sequence of control laws for the robot, $\sigma = u_0, u_1, \dots, u_T \in \mathcal{U}^T$ from time $t = 0$ to $t = T$ to minimize trace of the covariance in the target's estimate at time $t = T$. That is,

$$\min_{u(t)} \max_{z(t)} \text{tr}(\Sigma_T) \quad (6)$$

such that,

$$\Sigma_{t+1} = \rho_{t+1}(\Sigma_t), \quad t = 0, 1, \dots, T-1$$

where $\rho_t(\cdot)$ is the Kalman Riccati equation [16].

The Riccati equation, $\rho(\cdot)$, maps the current covariance matrix $\hat{\Sigma}_k$, under a new measurement to the covariance matrix at the next time step,

$$\rho_i(\hat{\Sigma}_k) = C_k \hat{\Sigma}_k C_k^T - C_k \hat{\Sigma}_k H_k^T (H_k \hat{\Sigma}_k H_k^T + \hat{\Sigma}_w)^{-1} H_k \hat{\Sigma}_k C_k^T + \Sigma_v \quad (7)$$

where H_k is the matrix of the measurement equation computed around $X_r(k)$ at time k . Σ_w is the covariance of the measurement noise given in Equations (3)–(5).

The true position of the target is unknown making it impossible to determine Σ_w exactly. Consequently, we use an estimate of Σ_w using the estimated target's position $\hat{X}_o(k)$. Therefore, the optimal solution for the problem defined will be a closed-loop policy that should map the estimated target's position to the optimal control action for the robot.

IV. CLOSED-LOOP CONTROL POLICY

References [14], [15] solve a similar problem but for a linear Gaussian system. The linearity assumption makes the Riccati equation independent of the position of the target. Consequently, they show an open loop policy can determine the optimal control sequence for the robot. In our case, the optimal control policy for this state-dependent observation model case will be an adaptive (closed-loop) control policy. However, this generalization comes at the expense of discretization of the set of possible target measurements. Specifically, we assume that the measurement at any time step is chosen from one of k tuples of candidate measurements. That is,

$$z(t) \in \{z_1(t), z_2(t), \dots, z_k(t)\}.$$

These candidate measurements can be obtained by, for example, sampling from the continuous distribution of zero

mean sensor noise around the current estimate of the target. For example, we can choose k candidate measurements from the data within 3 standard deviation of the mean value, which contain 99.7% of the possible measurements.

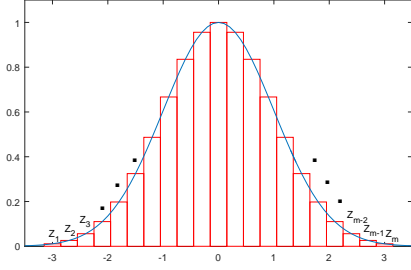


Fig. 1. Obtaining a finite set of candidate measurements by discretizing the Gaussian distribution of the measurement noise.

A. Optimal decisions: The minimax algorithm

Minimizing the maximum trace can be thought of as playing a game against an adversary: The robot chooses the control actions to minimize the trace whereas the adversary (*i.e.*, nature) chooses measurement noise to maximize the trace. By optimizing the min max trace, the robot determines the best conservative policy.

We can find this optimal strategy by building a minimax tree. This tree enumerates all possible control laws and all possible measurements that the robot can obtain. A node on the k th level of the tree stores the position of the robot, $X_r(k)$, the estimated position of the target, $\hat{X}_o(k)$, and the covariance matrix $\hat{\Sigma}_k$. Each node at an odd level has one branch per control action. Each node at an even level has one branch per candidate measurement. The robot's state and the target's estimate are updated appropriately along the control and measurement branches using the state transition equation (1) and the Kalman filter update equation, respectively. The minimax value is computed at the leaf nodes and is equal to the trace of the covariance matrix at that node. These values are propagated upwards to compute the optimal strategy. Figure 2 shows an example.

B. Alpha Pruning

The number of nodes in a naive minmax tree is exponential in the depth of the tree (*i.e.*, in the planning horizon). As a first step in reducing the size of the tree, we implement α pruning [13]. The main idea in α pruning is that if we have explored a part of the tree, we have an upper bound on the optimal minimax value. When exploring a new node, n_i , if we find that the minimax value of the subtree rooted at n_i is greater than the upper bound found, that subtree does not need to be explored further. This is because an optimal strategy will never prefer a strategy that passes through n_i since there exists a better control policy in another part of the tree. Note that n_i must be a control node. Measurement nodes cannot be pruned since the robot has no control over the actual measurement values. The pruning algorithm is based on the general alpha-beta pruning [13] used in adversarial

Algorithm 1: The minimax algorithms

```

1  $S_0 \leftarrow \{(X_r(o), \Sigma_0)\}$ ,  $S_t \leftarrow \phi$  for  $t = 1, \dots, T$ 
2  $\mathcal{Z} = \{z_1, z_2, \dots, z_k\}$ 
3 for  $t = 1 : T$  do
4   if NODE STATE (min)
5     for all  $(X_r(t-1), \Sigma(t-1)) \in S_{t-1}$  do
6       for all  $u_i \in \mathcal{U}$ 
7          $X_r(t) \leftarrow f(X_r(t-1), u_i)$ 
8          $S_t \leftarrow S_t \cup \{(X_r(t), \Sigma(t-1))\}$ 
9     else if NODE STATE (max)
10      for all  $(X_r(t), \Sigma(t-1)) \in S_t$  do
11        for all  $z_i \in \mathcal{Z}$ 
12           $\Sigma(t) \leftarrow \rho(X_r(t), z_i, \Sigma(t-1))$ 
13           $S_t \leftarrow S_t \cup \{(X_r(t), \Sigma(t))\}$ 
14 for  $t = 1 : T$  do
15   if TERMINAL-TEST (Max)
16     for each  $S_t$  do
17        $\mathcal{V} \leftarrow (\max \text{tr}(\Sigma_i(t)), S_t(i))$ 
18        $t \leftarrow t - 1$ 
19   if TERMINAL-TEST (Min)
20     for each  $S_t$  do
21        $\mathcal{V} \leftarrow (\min \text{tr}(\Sigma_i(t)), S_t(i))$ 
22        $t \leftarrow t - 1$ 
23 return  $\mathcal{V}$ 

```

games. Fig. 2 shows a simple example of policy tree built using alpha pruning.

C. Algebraic Redundancy Pruning

In addition to alpha pruning, we extend the ideas presented by [14] for linear systems and extend them to get even further computational savings. If there are multiple nodes at the same level with the same $X_r(t)$ values but different target estimates, we can prune one of the nodes if it is clearly “dominated” by the others. The notion of domination encodes the property that that particular node will never be a part of an optimal (minmax) policy. Reference [14] formalized the notion of domination by defining an algebraic redundancy constraint. We adapt this result for our notation as follows:

Theorem 1 (Algebraic Redundancy [14]): Let $\mathcal{H} = \{(X_r^j(t), \Sigma_t^j)\}$ be a set of n nodes at the same level of the tree. If there exist non-negative constants $\alpha_1, \alpha_2, \dots, \alpha_k$ such that,

$$\Sigma_t^p \succeq \sum_{i=1}^k \alpha_i \Sigma_t^i \quad \text{and} \quad \sum_{i=1}^k \alpha_i = 1$$

then the node $(X_r^p(t), \Sigma_t^p)$ is regarded as algebraically redundant¹ with respect to $\mathcal{H} \setminus \{(X_r^p(t), \Sigma_t^p)\}$ and $(X_r^p(t), \Sigma_t^p)$ and all of its descendants can be pruned without eliminating the optimal solution from the tree.

They prove that the trace of any successor of $(X_r^p(t), \Sigma_t^p)$ cannot be lower than one of the successors of

¹ $M \succeq N$ represents that $M - N$ is positive semi-definite.

$\mathcal{H} \setminus \{(X_r^p(t), \Sigma^p(t))\}$. Our main insight is that a similar redundancy constrained can be defined for the state-dependent case with suitable additional constraints as described below.

Theorem 2 (State-dependent Algebraic Redundancy): Let $\mathcal{H} = \{(X_r^i(t), \hat{X}_o^i(t), \hat{\Sigma}_t^i)\}$ be a set of N nodes at the same level. If there exists a node $A = (X_r^A(t), \hat{X}_o^A(t), \hat{\Sigma}_t^A)$ at the same level such that:

- 1) the robot states are identical, i.e., $X_r^A(t) = X_r^i(t)$ for all i in \mathcal{H} ;
- 2) the least common ancestor of A with any other node in \mathcal{H} is a control (i.e., min) node;
- 3) there exist non-negative α_i such that for any K :

$$H_t \Sigma_t^A H_t^T \succeq \sum_{i=1}^N \alpha_i [H_t \Sigma_t^i H_t^T + K (\delta_1^2 + \delta_2^2 \mathcal{C})] \quad (8)$$

where, $\sum_{i=1}^N \alpha_i = 1$, then there exists a node in \mathcal{H} , say B , such that:

$$\text{tr}(\Sigma_{t+K}^A) \geq \text{tr}(\Sigma_{t+K}^B).$$

That is, the node A can be pruned from the minimax tree without eliminating the optimal policy.

The proof is presented in the appendix.

D. Sub-optimal Pruning algorithm

Combining alpha pruning with linear state-independent algebraic redundancy pruning we can reduce a significant number of branches in the search tree while still guaranteeing optimality. We can further reduce the number of branches at the expense of losing optimality. This can be achieved by relaxing alpha-pruning and algebraic redundancy constraints. We use two parameters $\epsilon_1 > 0$ and $\epsilon_2 > 0$ as relaxation parameters for alpha pruning and algebraic redundancy pruning, respectively. In each case, we bound the loss in optimality as a function of the parameters.

Specifically, while building the tree, we prune away a node if it satisfies either of the following two conditions. When checking for alpha pruning, we prune a node if its alpha value is greater than or equal to the best minmax value found so far minus ϵ_1 . Similarly, we replace the constraint in Theorem 2 with the following:

$$H_t (\Sigma_t^A + \epsilon_2) H_t^T \succeq \sum_{i=1}^N \alpha_i [H_t \Sigma_t^i H_t^T + K (\delta_1^2 + \delta_2^2 \mathcal{C})] \quad (9)$$

By varying ϵ_1 and ϵ_2 , we can vary the number of nodes in the search tree. Next we bound the resulting loss in the optimality of the algorithm.

V. ERROR ANALYSIS

In this section, we bound the value returned by the relaxed algorithm with respect to the optimal algorithm.

Theorem 3 (ϵ_1 Alpha Pruning): Let $J_{2k}^* = \text{tr}(\hat{\Sigma}_{2k}^*)$ be the optimal minimax value returned by the full enumeration tree. If $J_{2k}^{\epsilon_1} = \text{tr}(\hat{\Sigma}_{2k}^{\epsilon_1})$ is the value returned by the ϵ_1 -alpha pruning algorithm, then $0 \leq J_{2k}^{\epsilon_1} - J_{2k}^* \leq \epsilon_1$.

The proof follows directly from the fact that if a node on the optimal policy, say n_i is pruned away, then the alpha value at n_i is at most the alpha value of some other node, say n_j , that is present in the tree minus ϵ_1 . The alpha value of n_j cannot be less than the value returned by the ϵ_1 algorithm. The full proof is given in the appendix. The bound for ϵ_2 -algebraic redundancy pruning is more complicated.

Theorem 4 (ϵ_2 State-dependent Algebraic Redundancy): Let $J_{2k}^* = \text{tr}(\hat{\Sigma}_{2k}^*)$ be the optimal minimax value returned by the full enumeration tree of $2k$ levels. If $J_{2k}^{\epsilon_2} = \text{tr}(\hat{\Sigma}_{2k}^{\epsilon_2})$ is the value returned by the ϵ_2 -algebraic redundancy pruning algorithm, then

$$0 \leq J_{2k}^{\epsilon_2} - J_{2k}^* \leq B^{\epsilon_2}$$

where,

$$B^{\epsilon_2} = \text{tr} \left\{ \sum_{j=0}^k \left[\prod_{i=k-1}^j (F_i(\Sigma) \Phi_{2i}(\Sigma)) \prod_{i=j}^{k-1} (F_i(\Sigma) \Phi_{2i}(\Sigma))^T \right] \epsilon_2 \right\}$$

where, $F_i(\Sigma) = C - CK_i(\Sigma)H_i$ and $K_i(\Sigma)$ is the Kalman gain given by $K_i(\Sigma) = \Sigma H_i^T (H_i \Sigma H_i^T + \Sigma_w)^{-1}$, and $\Phi_{2k}(\cdot)$ is the application of the Riccati equation $\rho(\cdot)$, over k measurement steps:

$$\Phi_{2k}(\cdot) = \underbrace{\rho_{2(k-1)}(\rho_{2(k-2)}(\dots \rho_0(\cdot)))}_{k \text{ steps } \rho(\cdot)}.$$

By combining the two results, we get

$$0 \leq J_{2k}^{\epsilon_1, \epsilon_2} - J_{2k}^* \leq \max \{\epsilon_1, B^{\epsilon_2}\}.$$

VI. SIMULATIONS

In this section, we present results from simulations to evaluate our pruning techniques. We carry out three types of evaluations. First, we investigate the savings of our algorithm by comparing the number of nodes in the pruned minimax tree and the full enumeration tree. Then, we study the effect of varying the ϵ_1 and ϵ_2 parameters on the number of nodes. Finally, we use the control policy given by our algorithm and execute it by drawing actual measurements from a random distribution. This represents a realistic scenario where the measurements are not necessarily adversarial. We demonstrate how our strategy can be used in such a case, and compare the average case performance with the worst-case performance.

In all simulations, the robot can choose from four actions:

$$\mathcal{U} = \{[+e, 0]^T, [-e, 0]^T, [0, +e]^T, [0, -e]^T\}$$

where e is a constant.

$$X_r(t+1) = X_r(t) + u(t)$$

We build the tree using five candidate measurements at each step: $z(t) = \{z_1(t), z_2(t), \dots, z_5(t)\}$. The five values are randomly generated with Gaussian noise.

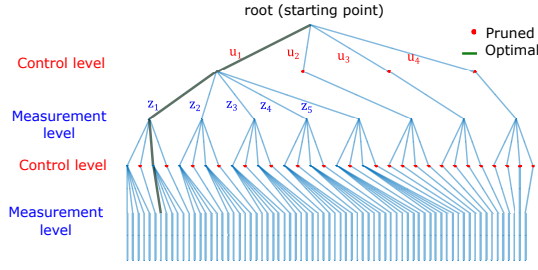


Fig. 3. A five level minimax tree with pruning (189 nodes) and full enumeration (505 nodes).

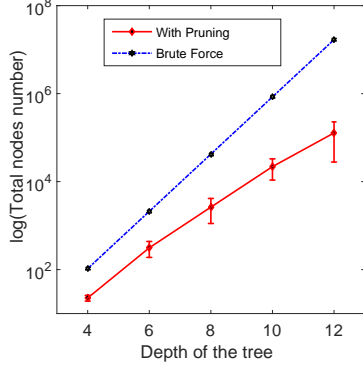


Fig. 4. Comparison of the number of total nodes generated for minimax tree. Note the log scale.

A. Comparing the Number of Nodes

Fig. 3 and Fig. 4 shows the number of nodes left in the minimax tree after pruning as compared to a full enumeration tree. We prune a node by comparing it to the nodes already explored. More nodes will be pruned if initial nodes encountered are “closer” to the optimal policy. For instance, if the first set of nodes explored happen to be control laws that drive the robot close to the target, then we expect the nodes encountered later will be pruned closer to the root. To provide a fair assessment, we generate the search trees for various true positions of the target and report the average and standard deviation of the number of nodes in Fig. 4.

Fig. 4 shows that our algorithm prunes orders of magnitudes of nodes from the full enumeration tree. For a tree with depth 13, it takes 8.08×10^7 to enumerate all nodes but the same optimal solution can be computed using 4.36×10^5 nodes with our pruning strategy.

Even though our algorithm prunes a large percentage of the nodes, the number of nodes still grows exponentially. By sacrificing optimality, we can prune even more nodes. We evaluate this by varying ϵ_1 and ϵ_2 individually first, and then jointly. As shown in Fig. 5, ϵ_1 -alpha pruning is relatively better at reducing the complexity of the minmax tree. This is intuitive because ϵ_1 -alpha pruning condition compares nearly every pair of nodes at the same depth. ϵ_2 -algebraic redundancy pruning, on the other hand, requires more conditions (same robot state) to be satisfied. Nevertheless, Fig. 5 shows that by sacrificing optimality, the number of nodes can be substantially reduced.

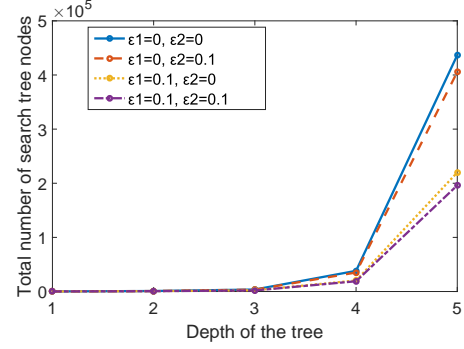


Fig. 5. Effect of the ϵ_1 and ϵ_2 relaxation parameters on the number of nodes in the search tree. The baseline case is the optimal solution with alpha pruning and algebraic redundancy with both parameters set to zero.

B. Online Execution of the Search Tree

So far, we have discussed the problem of building the minimax tree. Once the tree is built, the robot can execute the optimal policy. At the root node, the robot executes the first control action along the optimal minmax path found. Next, the robot obtains a measurement. This measurement may not correspond to the worst-case measurement. Furthermore, the actual value of the measurement may not even be in the k candidate measurements in the tree. The updated target estimate may not correspond to a node in the tree. Instead, we compute the distance between the actual measurement and find the closest match in the candidate set. The corresponding child node is now the new root node of the tree and the optimal policy starting at that node is executed, iteratively. Fig. 6 shows the execution in a simple instance.

VII. CONCLUSION

We investigated the problem of devising closed-loop control policies for state-dependent target tracking. Unlike state-independent tracking, the value of a candidate control law in state-dependent target tracking is a function of the history of measurements obtained. Consequently, planning over a horizon requires taking into account possible measurement values. In this paper, we focused on minimizing the worst-case uncertainty. Our solution consisted of building a min max search tree to obtain the control policy. A full enumeration tree has size exponential in the number of measurements, control laws, and the planning horizon. Instead, we exploited the structural properties of Kalman filter to yield a tree with significantly less number of possible nodes without sacrificing the optimality guarantees. We also showed how two parameters, ϵ_1 and ϵ_2 , can be used to yield even more computational savings at the expense of optimality.

One disadvantage of the generalization is the need to discretize the set of possible future measurements. Our immediate future work is to bound the suboptimality as a function of the number of discrete samples chosen to represent the continuous set of future measurements. Once a bound is obtained, the user may choose the correct trade-off between the computation time and the solution quality

desired. A second avenue of future work focuses on extending these results to multi-robot, multi-target scenarios. Our prior work [17] has shown a greedy assignment of robot trajectories to targets yield provably approximate solutions for one-step planning. We will extend this to planning over a finite horizon using the results presented in this paper.

REFERENCES

- [1] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation: Theory, Algorithms, and Software*. John Wiley & Sons, 2004.
- [2] X. R. Li and V. P. Jilkov, "Survey of maneuvering target tracking. part i. dynamic models," *IEEE Transactions on aerospace and electronic systems*, vol. 39, no. 4, pp. 1333–1364, 2003.
- [3] —, "Survey of maneuvering target tracking. part ii: motion models of ballistic and space targets," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 46, no. 1, pp. 96–119, 2010.
- [4] —, "A Survey of Maneuvering Target TrackingPart III: Measurement Models," in *Proceedings of SPIE*, vol. 4473, 2001, p. 424.
- [5] —, "A Survey of Maneuvering Target TrackingPart IV: Decision-Based Methods," in *Proceedings of SPIE*, vol. 4728, 2002, p. 512.
- [6] —, "Survey of maneuvering target tracking. part v. multiple-model methods," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 41, no. 4, pp. 1255–1321, 2005.
- [7] B. Rao, H. F. Durrant-Whyte, and J. Sheen, "A fully decentralized multi-sensor system for tracking and surveillance," *The International Journal of Robotics Research*, vol. 12, no. 1, pp. 20–44, 1993.
- [8] N. Karnad and V. Isler, "Modeling human motion patterns for multi-robot planning," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, 2012, pp. 3161–3166.
- [9] M. Montemerlo, J. Pineau, N. Roy, S. Thrun, and V. Verma, "Experiences with a mobile robotic guide for the elderly," in *Proceedings of the AAAI National Conference on Artificial Intelligence*, 2002, pp. 587–592.
- [10] V. Isler, N. Noori, P. Plonski, A. Renzaglia, P. Tokekar, and J. Vander Hook, "Finding and tracking targets in the wild: Algorithms and field deployments," in *International Symposium on Safety, Security, and Rescue Robotics*, 2015.
- [11] P. Tokekar, E. Branson, J. Vander Hook, and V. Isler, "Tracking aquatic invaders: Autonomous robots for invasive fish," *IEEE Robotics and Automation Magazine*, 2013.
- [12] P. Tokekar, J. Vander Hook, and V. Isler, "Active target localization for bearing based robotic telemetry," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011.
- [13] S. Russell and P. Norvig, *Artificial Intelligence: A modern approach*. Prentice-Hall, 1995.
- [14] M. P. Vitus, W. Zhang, A. Abate, J. Hu, and C. J. Tomlin, "On efficient sensor scheduling for linear dynamical systems," *Automatica*, vol. 48, no. 10, pp. 2482–2493, October 2012.
- [15] N. Atanasov, J. Le Ny, K. Daniilidis, and G. J. Pappas, "Information acquisition with sensing robots: Algorithms and error bounds," in *Proceedings of IEEE International Conference on Robotics and Automation*, 2014, pp. 6447–6454.
- [16] P. R. Kumar and P. Varaiya, *Stochastic systems: Estimation, identification, and adaptive control*. Prentice Hall, NJ, 1986, vol. 986.
- [17] P. Tokekar, V. Isler, and A. Franchi, "Multi-target visual tracking with aerial robots," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014.

APPENDIX

Theorem 5: (Monotonicity of state dependent Riccati equation)

If two nodes of a linear stochastic system satisfy the condition: $H\Sigma_t^A H^T + S^A \succeq H\Sigma_t^B H^T + S^B$ and $\Sigma_t^A \succeq \Sigma_t^B$, then after apply one step Riccati map, we have:

$$\rho(\Sigma_t^A) \succeq \rho(\Sigma_t^B)$$

where,

$$0 \leq S^A \leq a, 0 \leq S^B \leq a$$

$$\rho(\Sigma_k) = C_k \Sigma_k C_k^T - C_k \hat{\Sigma}_k H_k^T (H_k \hat{\Sigma}_k H_k^T + S^A)^{-1} H_k \hat{\Sigma}_k C_k^T + \Sigma_v \quad (10)$$

Proof:

$$\begin{aligned} & L(\Sigma_k^A) - L(\Sigma_k^B) \\ &= C_k \Sigma_k^A C_k^T - C_k \Sigma_k^A H^T (H \Sigma_k^A H^T + S^A)^{-1} H \Sigma_k^A C_k^T \\ & \quad - C_k \Sigma_k^B C_k^T + C_k \Sigma_k^B H^T (H \Sigma_k^B H^T + S^B)^{-1} H \Sigma_k^B C_k^T \end{aligned} \quad (11)$$

We define:

$$K(\Sigma) := -F \Sigma H^T (H \Sigma H^T + S)^{-1}$$

$$F(\Sigma) := F - (F \Sigma H^T) (H \Sigma H^T + S)^{-1}$$

Note that $F(\Sigma) = F + K(\Sigma)H$, and

$$K(\Sigma)(F \Sigma H^T)^T = -K(\Sigma)(H \Sigma H^T + S)K(\Sigma)^T$$

Then,

$$\begin{aligned} & L(\Sigma_t^A) - L(\Sigma_t^B) - F(\Sigma_t^A)(\Sigma_t^A - \Sigma_t^B)F^T(\Sigma_t^A) \\ &= K(\Sigma_t^A)F\Sigma_t^A H^T - K(\Sigma_t^B)F\Sigma_t^B H^T \\ & \quad - K(\Sigma_t^A)H(\Sigma_t^A - \Sigma_t^B)F^T - F(\Sigma_t^A - \Sigma_t^B)H^T K^T(\Sigma_t^A) \\ & \quad - K(\Sigma_t^A)H(\Sigma_t^A - \Sigma_t^B)H^T K(\Sigma_t^A)^T \\ &= K(\Sigma_t^A)F\Sigma_t^A H^T - K(\Sigma_t^B)F\Sigma_t^B H^T \\ & \quad - K(\Sigma_t^A)[H\Sigma_t^A F^T - H\Sigma_t^B F^T] \\ & \quad - [F\Sigma_t^A H^T - F\Sigma_t^B H^T]K^T(\Sigma_t^A) \\ & \quad - K(\Sigma_t^A)H(\Sigma_t^A - \Sigma_t^B)H^T K^T(\Sigma_t^A) \\ &= -K(\Sigma_t^B)(F\Sigma_t^B H^T)^T + K(\Sigma_t^A)(F\Sigma_t^B H^T) \\ & \quad + (F\Sigma_t^B H^T)^T K^T(\Sigma_t^A) \\ & \quad + K(\Sigma_t^A)(H\Sigma_t^B H^T + S^A)K^T(\Sigma_t^A) \\ &= K(\Sigma_t^B)(H\Sigma_t^B H^T + S^B)K^T(\Sigma_t^B) \\ & \quad + K(\Sigma_t^A)(H\Sigma_t^B H^T + S^A)K^T(\Sigma_t^A) \\ & \quad - K(\Sigma_t^A)(H\Sigma_t^B H^T + S^B)K^T(\Sigma_t^B) \\ & \quad - K(\Sigma_t^B)(H\Sigma_t^B H^T + S^B)K^T(\Sigma_t^A) \\ &= (K(\Sigma_t^B) - K(\Sigma_t^A))(H\Sigma_t^B H^T + S^B)(K(\Sigma_t^B) - K(\Sigma_t^A))^T \\ & \quad + K(\Sigma_t^A)(S^A - S^B)K^T(\Sigma_t^A) \end{aligned} \quad (12)$$

■

Define

$$\begin{aligned} M = & \max((F + K(\Sigma_t^A)H)(F + K(\Sigma_t^A)H)^T, K(\Sigma_t^A)K^T(\Sigma_t^A)) \end{aligned} \quad (13)$$

So, we have,

$$\begin{aligned}
& L(\Sigma_t^A) - L(\Sigma_t^B) \\
&= F(\Sigma_t^A)(\Sigma_t^A - \Sigma_t^B)F^T(\Sigma_t^A) + K(\Sigma_t^A)(S^A - S^B)K^T(\Sigma_t^A) \\
&\quad + (K(\Sigma_t^B) - K(\Sigma_t^A))(H\Sigma_t^B H^T + S^B)(K(\Sigma_t^B)^T - K(\Sigma_t^A)^T) \\
&= (F + K(\Sigma_t^A)H)(\Sigma_t^A - \Sigma_t^B)(F + K(\Sigma_t^A)H)^T \\
&\quad + K(\Sigma_t^A)(S^A - S^B)K^T(\Sigma_t^A) \\
&\quad + (K(\Sigma_t^B) - K(\Sigma_t^A))(H\Sigma_t^B H^T + S^B)(K(\Sigma_t^B) - K(\Sigma_t^A))^T \\
&\text{From [13], we have} \\
&\succeq K(\Sigma_t^A)H(\Sigma_t^A - \Sigma_t^B)H^T K^T(\Sigma_t^A) + K(\Sigma_t^A)(S^A - S^B)K^T(\Sigma_t^A) \\
&\quad + (K(\Sigma_t^B) - K(\Sigma_t^A))(H\Sigma_t^B H^T + S^B)(K(\Sigma_t^B) - K(\Sigma_t^A))^T \\
&= K(\Sigma_t^A)((H\Sigma_t^A H^T + S^A) - (H\Sigma_t^B H^T + S^B))K^T(\Sigma_t^A) \\
&\quad + (K(\Sigma_t^B) - K(\Sigma_t^A))(H\Sigma_t^B H^T + S^B)(K(\Sigma_t^B) - K(\Sigma_t^A))^T \\
&\quad \text{since } H\Sigma_t^A H^T + S^A \succeq H\Sigma_t^B H^T + S^B \\
&\succeq 0
\end{aligned} \tag{14}$$

PROOF OF THEOREM 2

Proof: We first prove a special case when \mathcal{H} consists of only one node, B . That is, we have:

$$H_t \Sigma_t^A H_t^T \succeq H_t \Sigma_t^B H_t^T + K \cdot aI$$

where, $a = \delta_1^2 + \delta_2^2 \mathcal{C}$.

To prove:

$$\text{tr}(\Sigma_{t+K}^A) \geq \text{tr}(\Sigma_{t+K}^B).$$

1) Show that the statement holds for $K = 1$.

When $K = 1$, $H\Sigma_t^A H^T \succeq H\Sigma_t^B H^T + aI$. From the Kalman Riccati map,

$$\begin{aligned}
& \Sigma_{t+1}^A = \rho_i(\Sigma_t^A, x_r(t), \hat{x}_o(t)) \\
&= C\Sigma_t^A C^T - C\Sigma_t^A H_t^T \\
&\quad (H_t \Sigma_t^A H_t^T + \Sigma_{w_i}(x_r(t), \hat{x}_o(t)))^{-1} H_t \Sigma_t^A C^T + \Sigma_v \\
&\text{Applying Theorem 5, we have} \\
&\succeq C\Sigma_t^B C^T - C\Sigma_t^B H_t^T \\
&\quad (H_t \Sigma_t^B H_t^T + \Sigma_{w_i}(x_r(t), \hat{x}_o(t)))^{-1} H_t \Sigma_t^B C^T + \Sigma_v \\
&= \Sigma_{t+1}^B
\end{aligned} \tag{15}$$

2) Inductive step: Show that if the claims holds for $K = M$, then it also holds for $K = M + 1$. This can be done as follows: Assume the claim holds for $K = M$. Let $\Sigma^{B'}(t) = \Sigma^B(t) + a \cdot (H^T H)^{-1}$, based on the condition of $K = M$ we have,

$$\Sigma_{t+M}^A \succeq \Sigma_{t+M}^{B'}$$

that is,

$$\Sigma_{t+M}^A \succeq \Sigma_{t+M}^B + a \cdot (H^T H)^{-1}$$

Similar to the step (1):

$$\Sigma_{t+M+1}^A \succeq \Sigma_{t+M+1}^B$$

Thereby showing that indeed $K = M + 1$ holds.

Since both the base case and the inductive step have been performed, by mathematical induction, the statement holds for all natural numbers n .

Then, we extend the proof from comparing two nodes to arbitrary N nodes case, if we have

$$H_t \Sigma_t^A H_t^T \succeq \sum_{i=1}^N \alpha_i [H_t \Sigma_t^i H_t^T + K(\delta_1^2 + \delta_2^2 \mathcal{C})] \tag{16}$$

Without loss of generality, we assume $\Sigma^{B'}$ is the minimum covariance matrix (in the positive semi-definite sense) among $\Sigma_t^1, \Sigma_t^2, \dots, \Sigma_t^N$.

$$\begin{aligned}
H_t \Sigma_t^A H_t^T &\succeq \sum_{i=1}^N \alpha_i [H_t \Sigma_t^i H_t^T + K(\delta_1^2 + \delta_2^2 \mathcal{C})] \\
&\succeq \sum_{i=1}^N \alpha_i [H_t \Sigma_t^{B'} H_t^T + K(\delta_1^2 + \delta_2^2 \mathcal{C})] \\
&= H_t \Sigma_t^{B'} H_t^T + K(\delta_1^2 + \delta_2^2 \mathcal{C})
\end{aligned} \tag{17}$$

Using the result from the induction above, after K minmax tree steps, there always exist a node B' , such that,

$$\text{tr}(\Sigma_{t+K}^A) \geq \text{tr}(\Sigma_{t+K}^{B'}) \tag{18}$$

Therefore, node A can be pruned without reducing the optimality of the minmax tree. ■

PROOF OF THEOREM 4

Proof: For some level i , suppose that we prune a node on the optimal policy. We have,

$$\text{tr}(H(\Sigma_{2i}^{\epsilon_2})H^T) \leq \text{tr}(H(\Sigma_{2i}^* + \epsilon_2 I)H^T)$$

From [14], we know that $\forall \Sigma, Q \in \mathbb{R}^{n \times n}$ and $\epsilon \geq 0$:

$$\rho_{2i}(\Sigma + \epsilon Q) \leq \rho_{2i}(\Sigma) + F_i(\Sigma)QF_i^T(\Sigma)\epsilon.$$

Applying to the above equation we get,

$$\begin{aligned}
\Phi_{2k}(\Sigma + \epsilon_2 Q) &= \rho_{2(k-1)}(\Phi_{2(k-1)}(\Sigma + \epsilon_2 Q)) \\
&= \rho_{2(k-1)}(\rho_{2(k-2)}(\dots \rho_0(\Sigma + \epsilon_2 Q))) \\
&\leq \Phi_{2k}(\Sigma) + \rho_{2(k-1)}(\rho_{2(k-2)}(\dots \rho_2(F_1(\Sigma)QF_1^T(\Sigma))) \\
&\vdots \\
&= \Phi_{2k}(\Sigma) + \\
&\quad \left[\prod_{i=k-1}^0 (F_i(\Sigma)\Phi_{2i}(\Sigma)) Q \prod_{i=0}^{k-1} (F_i(\Sigma)\Phi_{2i}(\Sigma))^T \right] \epsilon_2 \\
&\quad + o(\epsilon_2) \\
&\leq \Phi_{2k}(\Sigma) + \\
&\quad \left[\prod_{i=k-1}^0 (F_i(\Sigma)\Phi_{2i}(\Sigma)) Q \prod_{i=0}^{k-1} (F_i(\Sigma)\Phi_{2i}(\Sigma))^T \right] \epsilon_2
\end{aligned}$$

Let $\{\hat{\Sigma}_i^*\}_{i=1}^k$ be the series of covariance matrices along the optimal minmax trajectory. Suppose that the sequence of covariance matrices along the optimal trajectory returned

by ϵ_2 -algebraic redundancy pruning algorithm is $\left\{\hat{\Sigma}_i^{\epsilon_2}\right\}_{i=1}^k$.
We get,

$$\hat{\Sigma}_i^{\epsilon_2} \preceq \hat{\Sigma}_i^* + \epsilon_2 I, \quad \forall i = 1, 2, \dots, k$$

By combining the two results, we obtain the desired bound:

$$\begin{aligned} 0 &\leq J_{2k}^{\epsilon_2} - J_{2k}^* = \text{tr}(\hat{\Sigma}_k^{\epsilon_2}) - \text{tr}(\hat{\Sigma}_k^*) \\ &\leq \text{tr} \left\{ \sum_{j=0}^k \left[\prod_{i=k-1}^j (F_i(\Sigma) \Phi_{2i}(\Sigma)) \prod_{i=j}^{k-1} (F_i(\Sigma) \Phi_{2i}(\Sigma))^T \right] \epsilon_2 \right\} \\ &= B^{\epsilon_2} \end{aligned}$$

■

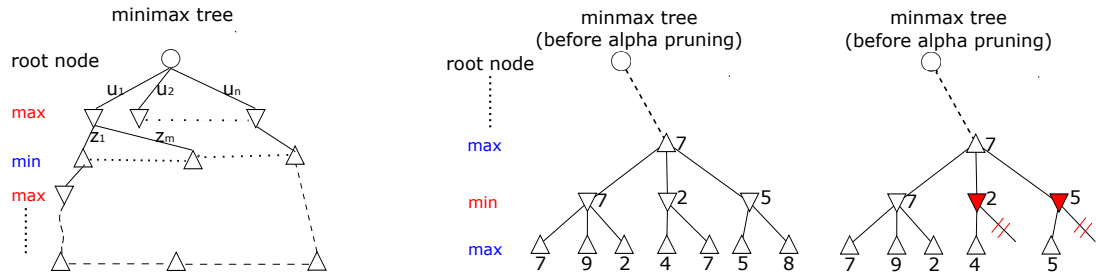


Fig. 2. A minimax tree with alpha pruning. ∇ and \triangle are nodes in which we compute the minimum or maximum value of its children. The value at the leaf nodes equals the $\text{tr}(\Sigma_k)$. ∇ and \triangle nodes represent control and measurement nodes, respectively. The filled ∇ are pruned by alpha pruning.

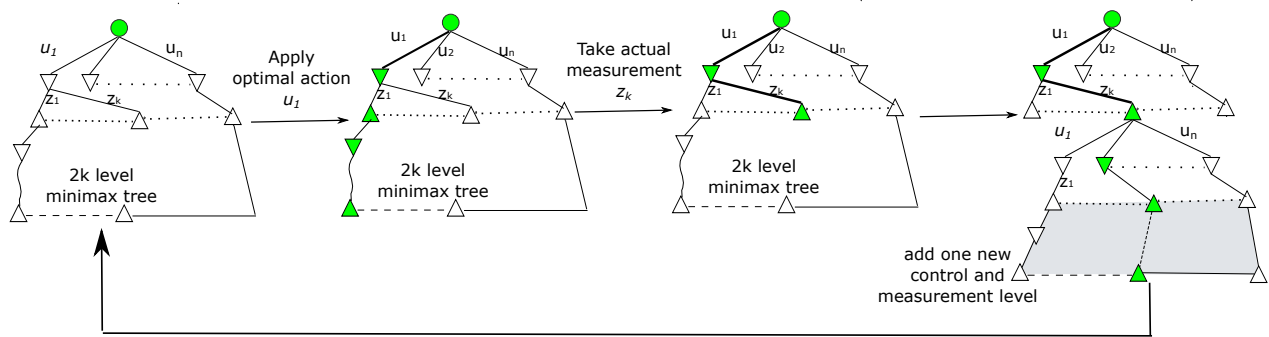


Fig. 6. Online measurement update with a minimax tree. The actual measurement, z_k may not correspond to a measurement node in the tree. In such a case, we choose the “closest” measurement in the tree.